Session 300

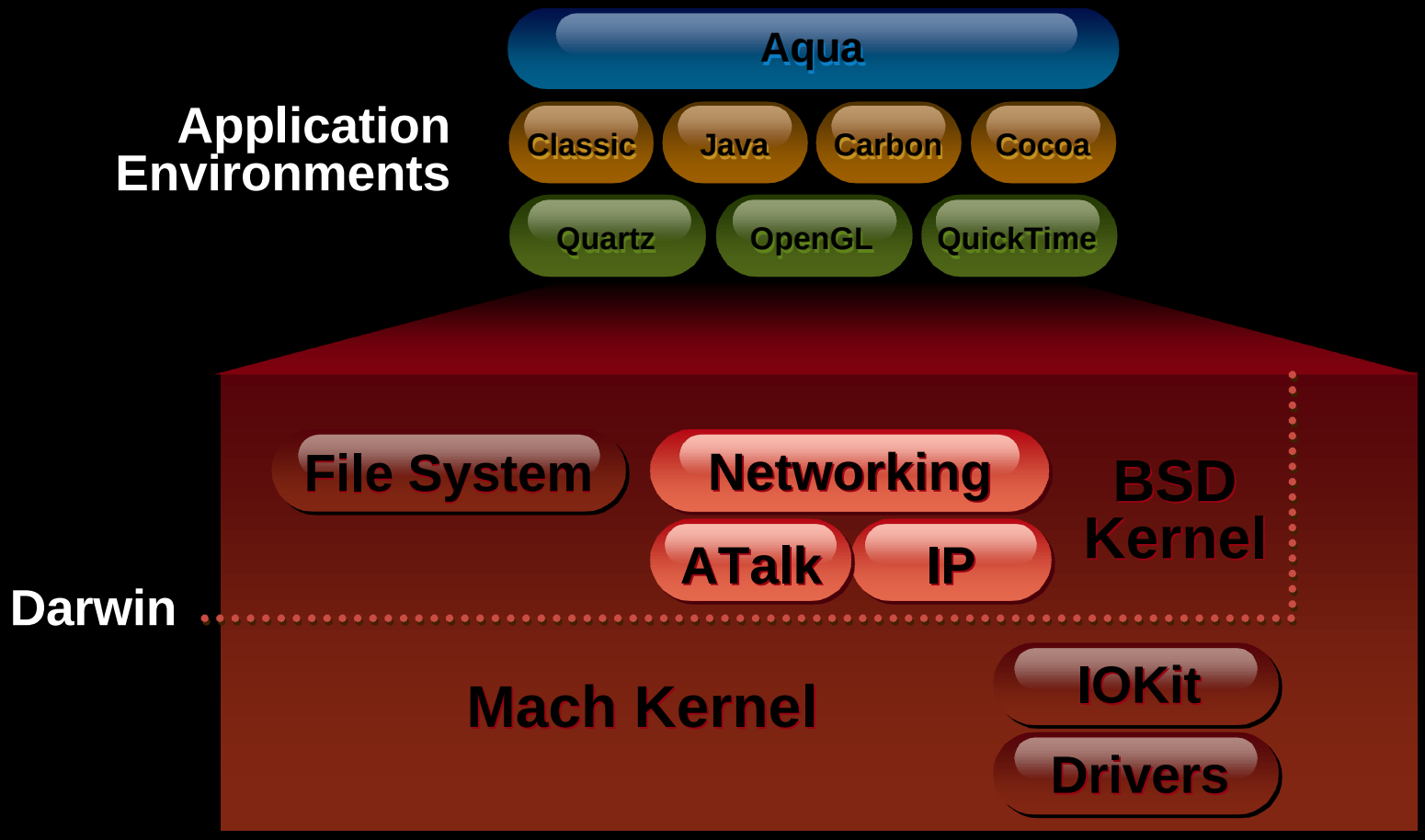
Mac OS Networking Overview

Vincent Lubet
Manager, Core OS Networking

Agenda

- Networking Overview for Mac OS X
- What are the APIs
- Hints and tips
- Future directions

Networking in Mac OS X



Networking in Mac OS X

Features

- TCP/IP and AppleTalk protocols
- Link layers
  - PPP with Serial and PPPoE
  - IOKit for Ethernet
- Dynamic configuration
- Extensible architecture

Networking in Mac OS X

Ongoing Goals

- Ease of use
- Performance
- Better extensibility
- Standards compliance

Networking in Mac OS X

Based on FreeBSD

- Currently based on FreeBSD 3.2
- Robust and proven implementation
- Popular API
- Easy to port Unix-like applications
  - Reuse available code

Networking in Mac OS X

Apple enhancement

- Multi-threaded and MP savvy kernel
- Tuned network buffer allocation
- Extensible—no need to recompile the kernel
- More in session 304, Networking in the Kernel

Dynamic Configuration

- No need to restart
- Supports multilink multihoming
- More in session 303, Network Configuration and Mobility

PPP

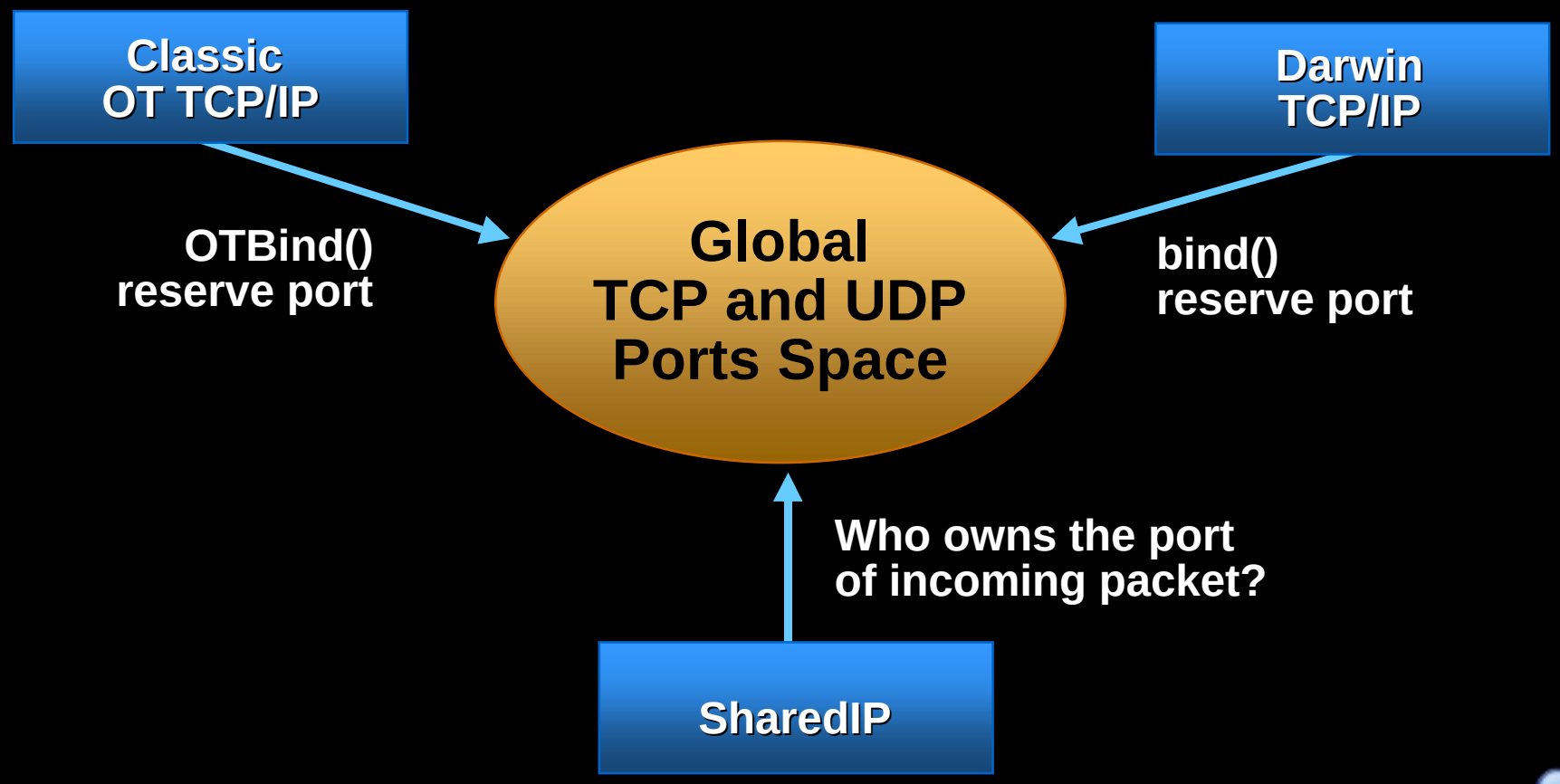
- Based on pppd
- Supports internal, external modems, PPPoE
- Apple enhancements
  - CCL scripts
  - OT/PPP control API

Classic Networking

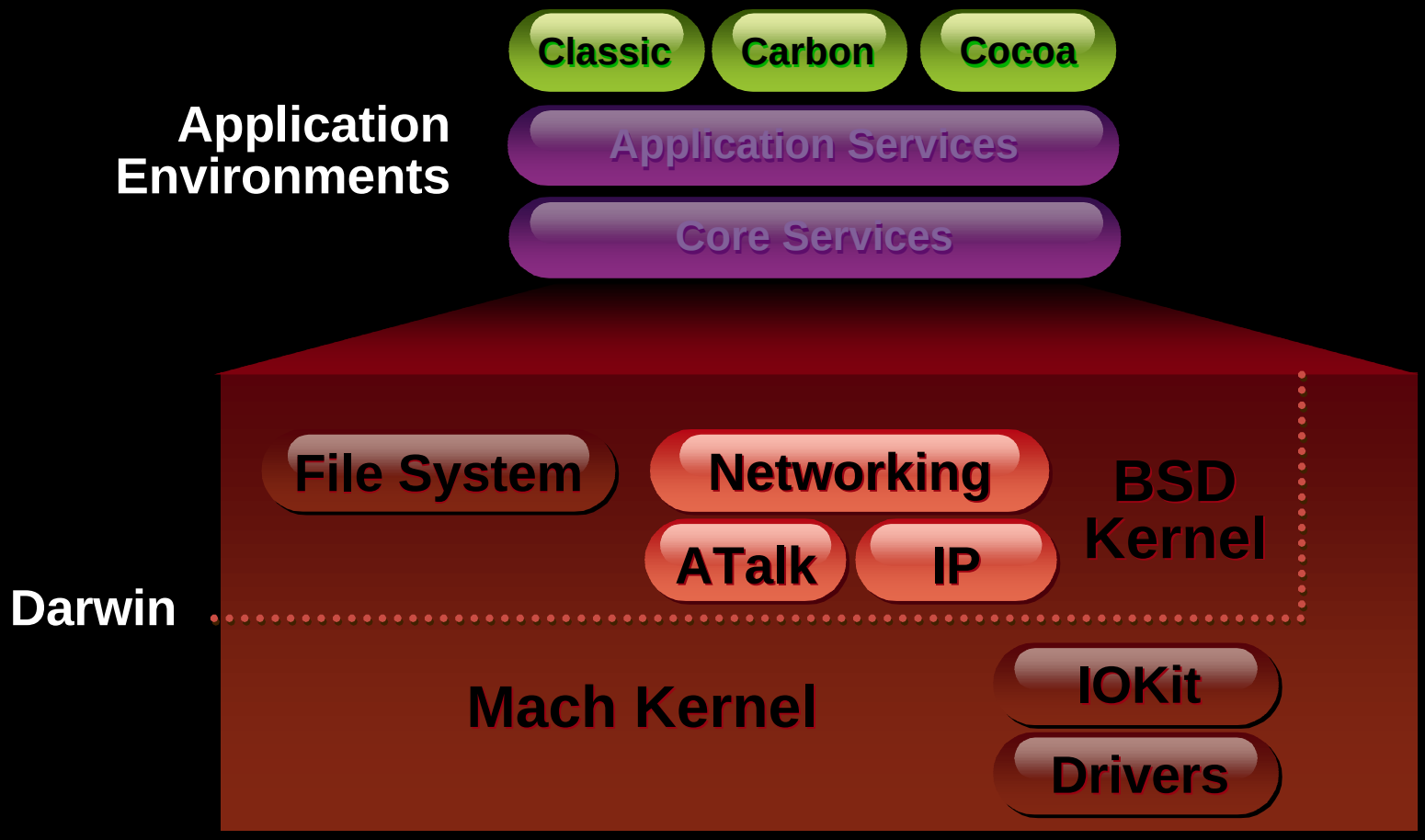
- Classic and Darwin share same IP address
  - TCP and UDP ports are shared
  - Single reply to ping!
- No user configuration, TCP/IP control panel read only
- Classic and Darwin have different AppleTalk addresses

Classic Networking

IP UDP and TCP port sharing



Networking in Mac OS X



Networking APIs

- URL oriented APIs
- Open Transport APIs in Carbon
- BSD sockets

URL Oriented APIs

- For non-network-centric applications
- Easy to download URLs
  - NSURLHandle: Cocoa
  - URLAccess: Carbon
  - CFURLAccess: Carbon and Cocoa
- More in session 311, Web Enabling Your Application

Carbon/OT

- Easy migration from Mac OS 8/9
- API level compatibility only
- OT implementation on Mac OS X
  - Framework built upon sockets
  - Uses threads to emulate asynchronous mode
  - About 5–10% overhead over straight sockets

Carbon/OT

Mac OS X and preemption

- Do not assume interrupt level nesting
  - Notifier can be preempted by main event loop
- Protect your data
  - OTEnterNotifier and OTLeaveNotifier to define critical sections
  - OT atomic operations for single variable access
  - OTLIFO for queues

Carbon/OT

Mac OS X tips

- The best ways to use OT
  - OTMP (not on Mac OS 8)
  - Synchronous blocking mode and sync idle events
  - Asynchronous mode within notifier
- Do not poll
- OTIdle adds latency

Carbon/OT

Mac OS X gotchas

- Flow control common on Mac OS X
  - Be ready to handle kOTFlowErr
  - Increase XTI_SNDBUF
- OT timer and deferred tasks
  - Not implemented with Deferred Task Manager and Time Manager
  - They are serialized with notifier

Carbon/OT

AppleTalk

- On Mac OS X OT is the only public AppleTalk API
- Supports the following protocols
  - DDP
  - ZIP
  - NBP

Sockets API

- Native API to get the most out of Mac OS X
- Lots of publicly available code
- Unix Network Programming by Stevens
- You can eliminate your sockets emulation layer

Sockets for OT Developers

Opening and connecting

Endpoint Provider OTOpenEndpoint()	Sockets file descriptor socket()
Active connection OTBind() -- qlen = 0 OTConnect()	bind() -- optional connect()
Passive connection OTBind() -- qlen > 0 OTListen() OTAccept()	bind() -- to specify port listen() -- backlog is qlen accept() -- return new file descriptor

Sockets for OT Developers

Sending and receiving

Sending data OTSnd() OTSnd() OTSndUData()	send() or sendmsg() sendto() or sendmsg()
Receiving data OTRcv() OTRcv() OTRcvUData()	recv() or recvmsg() recvfrom() or recvmsg()

Sockets for OT Developers

Disconnecting and closing

Abortive disconnect OTSndDisconnect()	setsockopt(SO_LINGER, abort)
Orderly disconnect OTSndOrderlyDisconnect() OTRcvOrderlyDisconnect()	shutdown(SHUT_WR) shutdown(SHUT_RD)
OTCloseProvider()	close()

Sockets for OT Developers

DNS

- Name to address resolution
  - gethostbyname()
- Address to name resolution
  - gethostbyaddr()
- Non-reentrant
- Blocking and not cancelable

Sockets for OT Developers

Tasking models

- Sync-blocking with threads
  - Use sockets equivalents
- Asynchronous
  - Convert to sync-blocking sockets
  - Deal with select()

Sockets for OT Developers

select() for multiplexing

- Wait for events among several file descriptors
- Three kinds of events: read, write, exception
- Thread blocks until timeout or at least one file is ready
- See Steven's book for details

Coding for Network Applications

Do not poll!

- Polling use 100 % of CPU
- Hurts other processes and uses more power
- Instead block and be event driven
- Use blocking threads
- Multiplex endpoints
  - For sockets use select()
  - For OT API use notifiers

Coding for Network Applications

Buffer sizes

- Send/receive sizes are critical for performance
  - Size of buffers passed to send/receive API
  - Too small = high context switch overhead
  - Too large = starves VM for buffers
- Things to tune
  - Socket buffer size
  - Size of buffer passed to API

Coding for Network Applications

Buffer sizes

- What is the correct buffer size?
  - Depends on bandwidth to destination
  - Depends on protocol, bulk transfers vs. transactions
  - Depends on number of clients
  - One size does not fit all
- Use large IO to file system
  - Buffer 32K before writing

Coding for Network Applications

Multihoming

- Computer does not have a single IP address
- IP addresses change over time
- For FTP client use **getsockname** or **OTGetProtAddress**
- Servers should bind to the any IP address
- Clients bind to nil

Coding for Network Applications

UI interaction

- Do not synchronize UI with network activity
- Use reasonable delays between update of status

The List

- Starting The List from scratch…

… we're listening to developer and user suggestions

Future

- IPv6 and IPSec
- ZeroConf
- More performance tuning
- Better extensibility
- Network setup API replacement

Additional Resources

Mac OS X

http://developer.apple.com/macosx

Classic Networking

http://developer.apple.com/macos/opentransport

Darwin

http://www.publicsource.apple.com

FreeBSD

http://www.freebsd.org

Related Sessions

300 Networking Overview	Room A1 **Tues., 9:00am**
304 Networking in the Kernel	Room J2 **Tues., 10:30am**
303 Network Configuration & Mobility	Room A1 **Tues., 5:00pm**
FF107 Networking Feedback Forum	Room J1 **Thurs., 10:30am**

Related Sessions (Cont.)

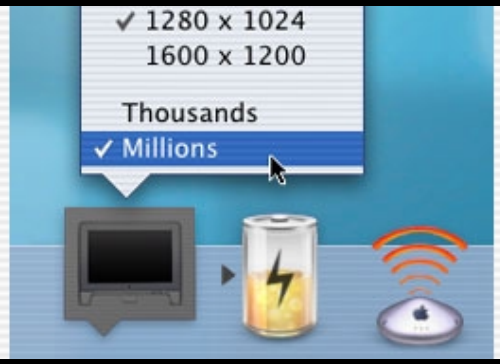
311 Web Enabling Your Application URLAccess and CFURLAccess	Civic **Thurs., 5:00pm**
136 Sound & Networking for Games Networking for Games Developers	Room J2 **Fri., 2:00pm**

Who to Contact

Vincent Lubet
Core OS Networking Manager
vlubet@apple.com

Thomas Weyer
Network & Communications Technology Manager
Apple Worldwide Developer Relations
weyer@apple.com

Q&A



Thomas Weyer
Network & Communications Technology Manager
weyer@apple.com

http://developer.apple.com/events/urls.html

WWDC

Worldwide Developers Conference 2001